Amendments to the Specification

Please replace the paragraph beginning at page 10, line 1, with the following rewritten paragraph:

A JIT compiler 108 can compile code as it is needed at runtime (just-in-time) from an architecture independent representation such as-CIL on a target execution architecture. For instance, a JIT compiler 108 built using interfaces from the SDA may [[to]] be designed for compiling CIL into native code at runtime on an AMD target execution architecture.

Please replace the paragraph beginning at page 10, line 6, with the following rewritten paragraph:

The defect detection tool 110 can [[be]] detect defects statically (before a program is run) in code written in one or more of languages. The defect detection tools can be constructed to run on any number of target execution architectures. For instance, a defect detection use 110 of the SDA may be to create a defect detection tool designed to take code written in the C++ programming language as input and detect defects in the code. The defect detection tool can be designed to execute on, for instance, an x86 target execution architecture.

Please replace the paragraph beginning at page 12, line 5, with the following rewritten paragraph:

FIG. 3 is a high-level block diagram of a system for creating components of software development tools using an SDA 300. One or more specification(s) 302 are created to describe target specific details a software development scenario. For instance, the specification(s) 302 might indicate which programming language the target tool may take as input, such as ~~Virtual~~ Visual Basic, C#, a .NET language such as CIL, or C++. Likewise, the specification(s) might indicate the target execution architecture for the target software development tool, such as the x86, AMD, MIPS, or IPF architecture. Additionally, the specification(s) may contain other information such as rule sets for type-checking the input language, or a set of class extension declarations for extensibly configuring the SDA core data structures. The specification(s) 302 is then used to create software development component(s) 304 that can be linked to the SDA 300. The linked component(s) 304 and SDA 300 may be used in combination to create any number of tools 306, such as a JIT compiler, a Pre-JIT compiler, a native compiler, or other software development tool. Alternatively, custom code may be added to implement additional functionality to the target software development tool 304.

Please replace the paragraph beginning at page 21, line 11, with the following rewritten paragraph:

In this embodiment, the IR used in an SDA can be a graph data structure organized as linear stream of dataflow operations represented by an operator, set of dataflow sources and set of dataflow destinations. Dataflow information and side-effects can be explicitly included in the dataflow sets. The IR may be strongly typed with source level type information to allow for robust code generation and type checking. The IR also may contain some or all of the dataflow and control flow links necessary to fully describe the program. In other words, if all the links necessary are present [[than]] then no other data structure is needed, such as a flow graph, or exception handling tree/graph.

Please replace the paragraph beginning at page 22, line 20, with the following rewritten paragraph:

The IR can be designed to directly support tree based and SSA based optimizations. This can be done by adding a def field on leaf tuples. [[It]] In its simplest form, expression temps can be linked together to form expression trees. This can be an invariant for pre-lowered code. This expression threading can be ignored by analyses that do not need to use the expression view. Transformations can leave them in a correct state and may use an expression optimizer for cleanup. Full blown SSA can use the same def fields and include special operations like a PHI instruction. Under SSA, the expression temps can simply become a subset of the overall SSA graph.

Please replace the paragraph beginning at page 46, line 19, with the following rewritten paragraph:

FIG. 25 illustrates an example of a computer system that serves as an operating environment for an embodiment of an SDA. The computer system includes a personal computer 2520, including a processing unit 2521, a system memory 2522, and a system bus [[2527]] 2523 that interconnects various system components including the system memory to the processing unit 2521. The system bus may comprise any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using a bus architecture such as PCI, VESA, Microchannel (MCA), ISA and EISA, to name a few. The system memory includes read only memory (ROM) 2524 and random access memory (RAM) 2525. A basic input/output system 2526 (BIOS), containing the basic routines that help to transfer information between elements within the personal computer 2520, such as during start-up, is stored in ROM 2524. The personal computer 2520 further includes a hard disk drive 2527, a magnetic disk drive 2528, e.g., to read from or write to a removable disk 2529, and an optical disk drive [[2570]] 2530, e.g., for reading a CD-ROM disk [[2571]] 2531 or to read from or write to other optical media. The hard disk drive 2527, magnetic disk drive 2528, and optical disk drive [[2570]] 2530 are connected to the system bus [[2527]] 2523 by a hard disk drive interface [[2572]] 2532, a magnetic disk drive interface [[2577]] 2533, and an optical drive interface [[2574]] 2534, respectively. The drives and their associated computer-readable media provide nonvolatile storage of data, data structures, computer-executable instructions (program code such as dynamic link libraries, and executable files), etc. for the personal computer 2520. Although the description of computer-readable media above refers to a hard disk, a removable magnetic disk and a CD, it can also include other types of media that are readable by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, and the like.

Please replace the paragraph beginning at page 47, line 15, with the following rewritten paragraph:

A number of program modules may be stored in the drives and RAM 2525, including an operating system [[2575]] 2535, one or more application programs [[2576]] 2536, other program modules [[2577]] 2537, and program data [[2578]] 2538. A user may enter commands and information into the personal computer 2520 through a keyboard 2540 and pointing device, such as a mouse 2542. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 2521 through a serial port interface [[2549]] 2546 that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port or a universal serial bus (USB). A monitor 2547 or other type of display device is also connected to the system bus [[2527]] 2523 via an interface, such as a display controller or video adapter 2548. In addition to the monitor, personal computers typically include other peripheral output devices (not shown), such as speakers and printers.

Please replace the paragraph beginning at page 48, line 10, with the following rewritten paragraph:

When used in a LAN networking environment, the personal computer 2520 is connected to the local network 2551 through a network interface or adapter [[2557]] 2553. When used in a WAN networking environment, the personal computer 2520 typically includes a modem 2554 or other means for establishing communications over the wide area network 2552, such as the Internet. The modem 2554, which may be internal or external, is connected to the system bus [[2527]] 2523 via the serial port interface 2546. In a networked environment, program modules depicted relative to the personal computer 2520, or portions thereof, may be stored in the remote memory storage device. The network connections shown are merely examples and other means of establishing a communications link between the computers may be used.